HEURISTIC ALGORITHMS FOR STATIC FLOW SHOP SCHEDULING PROBLEMS

A Thesis Submitted
in partial Fulfilment of the Requirements
for the Degree of
MASTER OF TECHNOLOGY

By
JAGDISH RAJ BHANDARI

to the

INTER-DISCIPLINARY PROGRAMME
IN
INDUSTRIAL AND MANAGEMENT ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY KANPUR

APRIL, 1976

To my wife, Anjana

CERTIFICATE

This is to certify that the present work on 'Heuristic Algorithms for Static Flow-Shop Scheduling Problems', by Jagdish Raj Bhandari has been carried out under my supervision and has not been submitted elsewhere for the award of a degree.

(J.L. Batra)
Convener
Industrial and Management Engg. Programme
Indian Institute of Technology, Kanpur

ACKNOWLEDGEMENT

It gives me great pleasure to express my indebtedness to Dr. J.L. Batra for having initiated me into this topic and for his continuous guidance and encouragement throughout the span of this work. I thank all my friends Messers P.N.Sarwade, J.K. Agrawal, R.K. Patil and R. Govindrajan for extending timely help in bringing out this manuscript.

Last but not the least, I thank Mr. J.K. Misra for the flawless typing of this transcript.

Jagdish Raj Bhandari

CONTENTS

LIST OF TABLES

SYNOPSIS

Jagdish Raj Bhandari, M.Tech. (Industrial and Management Engineering Programme), IIT Kanpur, April, 1976,
'Heuristic Algorithms for Static Flow Shop Scheduling Problems'.

The present work deals with static flow-shop scheduling problem. This problem has been formulated mathematically with the objective of minimizing the make-span time. Two heuristic approaches have been developed. Both the heuristic approaches generate schedules by sorting of certain functions. These functions pertain to six different strategies in heuristic one and three different strategies in heuristic two. All the strategies are based upon job and machine availability times of nodes. For a N-job problem heuristic one generates N-schedules, schedule giving minimum make-span is selected. Heuristic two generates only one schedule. All the strategies of both the heuristic algorithms have been evaluated for 240 randomly generated problems. Strategy one of heuristic one turns out to be the best.

A set of 42 problems with known optimum solution have been taken from literature and solved by heuristic algorithm one employing strategy one. It is found that the proposed algorithm (heuristic one with strategy one) which is computationally very efficient yields optimum or near optimum schedules in most of the cases.

CHAPTER I

INTRODUCTION

Everyone schedules in some fashion every day. Accordingly, scheduling means many things to many people. To a commuter, it brings to mind the train schedule; a machinist or pipefitter might see it as a list of materials to be used in the production of an item; a student would see it as a list of his classes. Scheduling is the allocation of resources over time to perform a collection of tasks. The practical problem of allocating resources over time to perform a collection of tasks arises in a variety of situations. The function of scheduling becomes relevant in a situation where the nature of the tasks to be scheduled has been described and the configuration of the resources available has been determined. The scheduling process most often arises in a situation where resource availabilities are essentially fixed by the long-term commitments of a prior planning decision.

Scheduling decisions are accomplished through four primary stages. The stages are: formulation, analysis, synthesis and evaluation. In the first stage, basically, the problem is identified and the criteria that should guide decision making are determined. This is often a subtle and complicated activity, but good decisions can seldom be expected without a

clear definition of the problem at hand and an explicit recognition of objectives. Analysis is the detailed process of examining the elements of a problem and their inter-relationships. This stage is aimed at identifying the decision variables and also at specifying the relationships among them and the constraints they must obey. Synthesis is the process of building alternative solutions to the problem. Its role is to characterize the feasible options that are available. Finally, evaluation is the process of comparing these feasible alternatives and selecting a desirable course of action. This selection is, of course, based on the criteria that were developed at the outset.

Scheduling approach begin with a translation of decision-making goals into an explicit objective function and decision-making restrictions into explicit constraints. Ideally, the objective function should consist of all costs in the system that depend on scheduling decisiens. In practice, however, such costs are often difficult to measure, or even to identify completely. Three types of decision-making goals seem to be prevalent in scheduling: efficient utilization of resources, rapid response to demands, and close conformance to prescribed deadlines. Frequently an important cost-related measure of system performance (such as machine idle time, job waiting time, make span, or job lateness) can be used as a substitute for total system cost.

Two kinds of feasibility constraints are commonly found in scheduling problems. First, there are limits on the capacity of available resources and, second, there are technological restrictions on the order in which tasks can be performed. A solution to a scheduling problem is any feasible resolution of these two types of constraints. Thus 'solving' a scheduling problem amounts to answering two kinds of questions:

1. Which resources will be allocated to perform each task?
2. When will each task be performed?

In other words, the essence of scheduling problems gives rise to (1) allocation decisions and (2) sequencing decisions.

The vital elements in scheduling models are resources and tasks. In the scheduling literature resources are typically characterized in terms of their qualitative and quantitative capabilities, so that a model describes the type and the amount of each resource. An individual task is described in terms of such information as its resource requirements, its duration, the time at which it may be started, and the time at which it is due. In addition, a collection of tasks may sometimes be described in terms of the technological constraints (precedence restrictions) that exist among its elements.

The amount of confusion resulting from various interpretations of the terms 'sequencing' and 'scheduling' is apparent in literature. Sequencing is concerned with the arrangements and permutations in which a set of jobs under consideration are performed on all machines. Scheduling is concerned with the specifications of the starting or completion times of certain jobs on all machines. Sequencing decisions focus on the arrangement of events; whereas, scheduling decisions focus on the time of events. The terms sequencing and scheduling, though distinguishable as above, are used frequently as synonyms. The reason for the frequent use of both terms interchangeably is because in practice, it is always assumed that each job is started as early as possible and therefore a schedule is automatically created when a particular sequence is sought.

As we attempt to structure the shop scheduling problem, a variety of situations encountered in operational systems dictate the types of scheduling models. For purposes of analysis, however, shop scheduling models may be distinguished by one or more of the following:

1. a single-machine process versus a multi-machine process;
2. a unique flow pattern for each job versus an identical flow pattern for all jobs;

3. a fixed finite number of jobs to be performed on various machines versus jobs arriving at the shop in a continuous fashion;
4. a complete and known information relative to jobs and machines versus one or more of the elements involved behaving in a probabilistic manner.

In the first class of models, the number of machines in the shop provides the distinction between single-machine and multi-machine problems. Single machine problems may seem to be trival or elementary for thorough analysis; however, their study is meaningful because there exist many operational systems of practical value in which a single-machine is involved. Also there is always a hope that any results in the single machine case may lead to new avenues of possible investigations in the multi-machine case.

In the second class of models; the characteristic of job flow pattern in a multi-machine process provides the distinction between flow-shop and job-shop problems. Each job in a job-shop has a different flow pattern. However, in a flow-shop there exists a single flow pattern in such a way that all jobs flow from one end of the shop to the other end. Thus machines can be thought of as existing in series. In such a shop the machines are normally numbered in ascending order starting from one end of the shop.

In the third class of models, the behavior of job arrivals at the shop provides the distinction between static and dynamic problems. In a static problem a fixed finite number of jobs arrive simultaneously at the shop and are immediately available to be performed on a number of idle machines. Thus the static problem does not explicitly take the variable time into consideration. On the other hand, in a dynamic problem jobs are arriving intermittently at the shop. Thus the dynamic problem deals with the variable time.

The static problem is, in itself, a drastic simplification of the realistic shop process. Analytical solutions thus far obtained for this problem are restricted to very simple cases and thus have a little practical value. However, the static problem is of interest as a preclude to the dynamic problem because it provides a way to treat the dynamic problem as a series of static problems.

In the fourth class of models, the behavior of the elements comprising the basic structure of the problem provides the distinction between deterministic and stochastic problems. The deterministic models are characterized by the fact that these elements do not involve chance variation and that the consequences of any given decision can be predicted in a precise manner. The stochastic models, however, are characterized by their explicit recognition of chance variations and uncertainty which could exist in one or more of these elements. In such a

case, the elements which vary stochastically, are assumed to be practicable only in a statistical sense.

Traditional scheduling procedures are typically designed to cope with the complexities of shop scheduling, where each job is unique and no prescribed technological ordering exists. These procedures depend on graphical devices such as Gantt chart or its concept to help maintain control of the shop operations. Essentially, Gantt chart is a device for portraying the information regarding the progress of each job in the shop and the current capacity of machines. It also pinpoints potential and actual problems in time for corrective action to be taken. Although Gantt charts are useful in developing feasible schedules, they are not, of course, a substitute for decision making.

Until 1954, there has been virtually no reported attempt to treat the shop scheduling problem analytically with the explicit purpose of optimizing some measure of performance. Since that time, however, the study of the shop scheduling problem and its contexts has attracted researchers from every field. This is due to the development of operations research techniques and their application to an area previously dominated by intuition and judgment; and due to the increased awareness of management to the potential gains realizable when good scheduling practices are employed.

The variety of situations that can be encountered in shop scheduling problems has been the object of considerable research efforts. Shop scheduling problems are analyzed by the application of sequencing and dispatching approaches. In the sequencing approach, a sequence of jobs for each machine is sought in advance. Thus, sequencing pertains to the process of specifying an arrangement of a set of jobs to be performed on each of the machine.

In the dispatching approach, on the other hand, dispatching rules are sought that can be applied at each machine to assign a job from among a number of jobs waiting to be performed. The selection of a dispatching rule must be made in the light of its effect on the operations in the shop. Examples of dispatching rules are: (1) select the job on first-come, first-served basis; (2) select the job with the shortest processing time; and (3) select the job with the earliest due-date. Thus dispatching pertains to the process of assigning a job to a particular machine according to some dispatching rule which is usually based on immediate or current shop information.

Theoretical models representing simple shop scheduling problems have been formulated and analysed using combinatorial analysis, mathematical programming, enumeration, Monte-Carlo sampling, and branch and bound approaches. Due to the combinatorial nature of sequencing problems, all the above mentioned

techniques are computationally less efficient for practical size problems. Such an attribute and its corresponding ramifications have been diluted through the employment of the high speed computer. However, it still remains that the shop scheduling problem commands a computational degree of severity that renders many approaches to its solution immobile.

In the present work two heuristic approaches have been developed for static flow shop problem. Both the heuristics approaches generate schedules by sorting of some functions. These functions are obtained by six different strategies in heuristic one and three different strategies in heuristic two. All the strategies are based upon job and machine availability times of nodes. For a N-job problem, heuristic one generates N-schedule, schedule giving minimum make-span is picked up. Heuristic two generates only one schedule. The results of various strategies for both the heuristics have been compared.

The potentialities of the proposed heuristics to generate optimal or near optimal schedules has been evaluated by comparing the results obtained from these heuristics for standard static flow shop problems for which optimal solutions are available in literature. A brief survey of the literature is presented in the next chapter.

CHAPTER II

LITERATURE SURVEY

Considerable research effort has been directed towards the solution of the combinatorial optimization problems associated with the scheduling of N-jobs through M-machines, the jobs having the same technological ordering. Minimization of make-span time (total elapsed time or maximum flow time) and average flow time are generally considered as the optimization criteria for this class of problems. The current approaches to solve the static flow-shop scheduling problems can be categorized broadly under the following headings:

1. Mathematical programming methods.
2. Enumerative and Monte-Carlo sampling techniques.
3. Branch and Bound and Elimination techniques.
4. Decomposition technique.
5. Heuristic approaches.

A brief description of these methods is given in the following paragraphs:

2.1 Mathematical Programming Methods.

Probably the most frequently cited paper in the field of scheduling is Johnson's solution to the two-machine flow-shop problem[30] He gives an algorithm for sequencing n jobs,

all simultaneously available, in a two-machine flow-shop, so as to minimize the maximum flow-time. All the jobs are to be processed first on machine A and then on machine B. He has also proposed an extension of his technique to handle the special three-machine cases in which either min $A_i \geqslant$ max B_i or min $D_i \geqslant$ max B_i; where A_i, B_i and D_i are the processing-time of the i-th job on first, second and third machine respectively. This paper is important, not only for its own content, but also for the influence it has had on subsequent work.

Bowman[11], Wagner[48], Giglio and Wagner[20] and Manne[35] have formulated the problem as an integer linear programme. Bowman estimates that formulating a simple problem involving three jobs and four machines by integer programming would involve 300 to 600 variables and many more constraints. The formulation by Wagner of the problem would be of the same order of magnitude. The solution of an integer programming problem by Gromory's cut algorithm, involves addition of still more constraints such that the final number of constraints would be considerably large.

Giglio and Wagner[20] have reported a linear programming formulation. They have solved the problem by the Simplex method and the resulting values are rounded to integers. According to them linear programming approach produces

solutions that are better as compared to the solutions obtained by sampling. However, the probability of generating optimal solution is low in the L.P. method. Further, they have reported that in the integer programming approach the convergence is not fast enough and also the memory locations required increases tremendously as the number of items to be sequenced increases.

Ashour and Char[2] have formulated the N-job, M-machine sequencing problem as a zero-one programme. They used Pseudo-Boolean approach to solve the resulting zero-one programme. For a three-job, three-machine problem the number of zero-one variables and constraints will be 33 and 9 respectively, while for a four-job, three-machine case the corresponding numbers will be 52 and 9, variables and constraints increase quadratically and linearly respectively with the number of jobs. Theoretically each zero-one programming problem can be solved. However, this procedure presents difficulties due to the control of computer round of error, computer storage and computational time. These drawbacks limit the applicability of this technique to small size problems only.

Dudek and Teuton[20] have developed a combinatorial approach for the multi-machine problem. They claim that their method gives optimal solutions. However, Karush[31] has refuted their claim by publishing a counter example. Smith and Dudek[43] have suggested a switch and check method which relies on changing

one partial sequence to another through switching the jobs around. Job and sequence dominance checks are used to eliminate the alternate partial sequences at each sequence position. Ashour[1] has reported that the computer time is more for switch- and check method than branch and bound method.

2.2 Enumerative and Monte-Carlo Sampling Techniques:

Giffler and Thompson[19] have proposed an enumerative procedure for handling static job-shop problems. Their algorithm systematically investigates a subset of feasible schedules. This subset contains the optimum. The optimal solution is found either by enumerating the subset or by sampling procedure depending on whether the subset is small or large. When the problem size is large there will be many sequences to be generated and evaluated by enumeration method, hence sampling procedures are introduced. In the sampling procedure, a number of feasible schedules are generated and the sequence with minimum makespan time is chosen as the best available sequence. Heller[28] studied the nature of distribution of elapsed times over the set of feasible sequences for a flow shop problem and found that the distribution tends to normality as the number of jobs become large. He observed that for finding minimum schedule time, one should sample from those schedules which have the same ordering on all machines.

2.3 Branch and Bound and Elimination Techniques:

The branch and bound algorithm developed by Little, et al[33] for the travelling salesman problem was the starting point in the development of many branch and bound algorithms for Flow-shop scheduling problem. Many researches have done a considerable amount of work in this area and have achieved some success. Notable contributions have been made by Ignall and Schrage[29], Brown and Lomnicki[13], Brooks and White[12], McMohan and Burton[36], Charlton[15], Nabeshima[38], and Gupta[25]. All these investigators have used the same philosophy for search in the use of branch and bound technique for solving scheduling problems. The efficiency of a branch and bound procedure depends mainly on the determination of the lower bounds. The bounding procedures proposed are based on either the jobs or the machines or both the jobs and machines.

In branch and bound procedure set of all feasible sequences is progressively divided into subsets in a systematic manner by 'branching'. After every division the newly created subsets must be mutually exclusive and exhaustive. A partial sequence (one with less than N jobs sequenced) is used to identify the subset of all possible sequences arising out of the partial sequence. A lower bound on each subset is determined such that the makespan for any sequence in the subset is greater than or equal to this bound. After every branching

the subset with the lowest lower bound is chosen for further exploration. The procedure terminates when a complete sequence is obtained for which the makespan is equal to or less than the lower bounds on all explored subsets. The complete sequence is then optimal. The branch and bound procedure can be represented by nodes on a tree.

Ignall and Schrage's[29] branch and bound algorithm is based on machine bounds which are dependent on the completion of all the jobs on the last machine. Brown and Lomnicki[13] have also proposed an algorithm based on machine bounds. The essential difference between the two approaches is that the bounds proposed by Brown and Lomnicki consider the idle time upto a machine caused by the augmentation of another job to the partial sequence, whereas this effect has not been considered by Ignall and Schrage.

McMohan and Burton[36] utilize a job based bounding procedure. They have proposed composite bounds combining the job based bounds and Ignall and Schrage's machine based bounds. They report that composite bounds are more efficient as compared to the machine based bounds.

Gupta[25] has proposed a modified composite method. The modification takes into account the idle time caused by a job when augmented to a partial sequence. Gupta claims that his method requires 30 percent less computational time to that of

Brown and Lomnicki's method. Brooks and White[12] used branch and bound approach in conjuction with the algorithm described by Giffler and Thompson[19] to solve N-job, M-machine scheduling problem. Brooks and White claim that their method gives optimal or near optimal solutions. Their method is based on the calculation of a job-based bound a machine-based bound. Charlton[15] has suggested a branch and bound method that exploits the graphical nature of the sequencing problem. Nabeshima[38] has presented an efficient method (less computation time) by using Johnson's criterion for the two machines problem. His method requires lesser number of evaluation of nodes as compared to the methods proposed by Lomnicki[34], and Ignall and Schrage[29]. As the number of jobs in the set increases, the computation time increases.

Recently, Baker[7] analysed Branch and Bound techniques for flow shop scheduling problem, and quoted that, composite bound is preferred, for, when the problems are easy, it is not much slower than other bounds, but when the problems are difficult it is often substantially faster than the other bounds. Moreover, its desirability would be even more decisive for larger problems.

Szwarc[47], Baker[8], and Gupta[26] have handled flow-shop problem by Elimination method (combinatorial approach). This approach utilizes special elimination conditions in order to construct a set of dominant schedules and then enumerates the

schedules in this dominant set in order to find an optimum. Gupta considered the objective function to be weighted sum of completion times of all jobs on all machines. He has stated that this approach is inappropriate and impractical as it cannot be used to solve flow-shop scheduling problems containing ten or more jobs.

Baker[7] has proposed a composite algorithm. This algorithm is composite of Branch and Bound method and elimination method. The branch and bound algorithm is applied to the dominant set. This reduces computational time as compared to when only branch and bound technique is utilized.

2.4 Decomposition Technique:

This approach has been proposed by Ashour[1,2,3,4]. In this method the original set of jobs is partitioned into a series of smaller, more manageable subsets. Each subset is solved by one of the existing techniques to determine its sequence. The scheduled times of the subsets are then combined to form a schedule time for all jobs of the original set. It was observed that the decomposition of the original problem into two subsets, each having N/2 jobs was more effective than other decomposition rules. Ashour claims that the quality of solutions produced by his technique is comparable with that of branch and bound method without back-tracking[1].

2.5 Heuristic Approaches:

The branch and bound approach and the elimination approach has two inevitable disadvantages. First, the computational requirements will be severe for large problems. Second, even for relatively small problems, there is no guarantee that the solution can be obtained quickly, since the extent of the partial enumeration depends on the data in the problem. Heuristic algorithms avoid these two drawbacks: they can obtain solutions to large problems with limited computational effort, and their computational requirements are predictable for problems of a given size. The drawback of heuristic approaches is, of course, that they do not guarantee optimality; and in some instances it may even be difficult to judge their effectiveness.

A number of heuristic modes have been proposed by many authors. The notable among them are: Giglio and Wagner, Ashour and Parker, Nabeshima, Krone and Steiglitz, Gupta, Palmer, Campbell-Dudek and Smith, Subrahmanyan, and Gupta and Maykut. Giglio and Wagner[20] applied Johnson's[30] procedure to general three-machine flow-shop problemswhere the processing-times were randomly generated and did not satisfy the condition $\min A_i \geqslant \max B_i$ or $\min D_i \geqslant \max B_i$ [30]. For twenty problems of six-job three-machine, the average maximum flow-time yielded by the procedure was 131.7, whereas the average optimal time was 127.9. Ashour and Parker[5] have developed an out-of-Kilter algorithm to deal with the machine sequencing problem. It yielded

first pass solutions of good quality with relatively inexpensive computational demands. Nabeshima's[39] approximate method is based on dynamic programming. Krone and Steiglitz[32] have proposed a two-phase heuristic algorithm, for the flow-shop scheduling problem, the objective being minimization of mean job-completion time.

Palmer[40] has suggested a slope index method, while Gupta[22] has reported a functional algorithm for the flow shop problem. Both of them have used the analogy between scheduling and sorting. The guideline suggested by Palmer can be stated qualitatively as follows: give priority to jobs having the strongest tendency to progress from short times to long times in the sequence of operations. Gupta's[22] functional algorithm is based on assigning a function to each job and then arranging the job in the ascending order of the function. The function proposed is an extension of Johnson's two-machine and three machine problems. Gupta claims that his method is superior to Palmer's slope index method. Campbell, Dudek and Smith's[14] algorithm is an extension of Johnson's model. For M-machine problem, this procedure generates (M-1) sequences and select the sequence which yields lowest total elapsed time. They have also compared their algorithm's effectiveness with Palmer's method. They claim that their method gives better results as compared to slope index method.

Subrahmanyam[46] has suggested three heuristic algorithms, all of which are extensions of Johnson's two machine model. He observed that the method of moments algorithm has an edge over the other two algorithms proposed by him. Subrahmanyam claims that his method of moments algorithm is superior to the algorithms proposed by Palmer, Gupta and Campbell et al.

CHAPTER III

PROBLEM FORMULATION AND SOLUTION METHODOLOGIES

The problem formulation and solution procedures for the static flow shop scheduling problem are presented in this chapter. A few basic terms like Job, Machine, Operation, Processing Time, Machine Order, Job Sequencing, Sequence, Make Span Time have been used in this chapter. The definitions of these basic terms are given in Appendix.

3.1 Statement of the Problem:

Given a set of jobs to be processed on a set of machines in the same order, the processing time of each job on each machine being given, find the order in which these jobs should be processed on the machines such that the make span time is minimum.

3.2 Assumptions:

The model developed are based on a set of assumptions. The purpose of these assumptions is to simplify the analysis of the problem and to increase generality of the models developed. The assumptions have been classified according to the characteristics of the jobs, machines, and processing times.

1. Assumptions Concerning Job Characteristics:

   1.1 Each job is processed according to a prespecified machine ordering, and no alternative ordering is permitted.

1.2 Each job, once started, must be performed to completion, that is, no job cancellation occurs.

1.3 Each job, once started on a machine, must be performed to completion before another job can start on that machine, that is, no pre-emptive priorities.

1.4 Each job is an entity, even though the job may be composed of individual units. This eliminates job splitting between two or more machines.

1.5 Each job may not be processed by more than one machine at a time.

1.6 Each job may have to wait between machines and hence, in-waiting inventory is permitted.

1.7 Each operation can be performed by only one machine.

1.8 Every job requires processing on every machine and no job is processed more than once by any machine. There will be (N x M) operations in a N-job, M-machine problem. If a certain job does not require processing on a particular machine, the corresponding operation will have a zero processing time.

1.9 All jobs are considered equal in importance. Thus there are no due dates, priorities, or rush orders.

2. Assumptions Concerning Machine Characteristics:

2.1 Each machine center consists of only one machine, i.e. there is only one machine of each type in the shop.

2.2 Each machine in the shop operates independently, and thus each machine is capable of operating at its own maximum rate of output.

2.3 Each machine is continuously available for assignment, during the scheduling period under consideration, without any interruption such as machine break-downs or maintenance.

2.4 Each machine can process at most one job at a time. This eliminates the machines which are designed to process more than one job at a time such as multi-spindle drills.

3. Assumptions Concerning Processing Time Characteristics:

3.1 Processing times are assumed to be known without error.

3.2 Set-up time and the time required to transport jobs between machines is zero.

3.3 Processing times are independent of the sequence in which the jobs are performed.

3.3 Nomenclature:

The following nomenclature is used to define the problem:

N = total number of jobs

M = total number of machines,

i = job number; $i = 1,2,\ldots, N$.
j = machine number; $j = 1,2,\ldots, M$.
t_{ij} = processing time for the i-th job on j-th machine

$(i = 1,2,\ldots,N;\ j = 1,2,\ldots, M)$.

$TJOB_i$ = sum of processing times for the i-th job on all the machines.

U_{ij} = job availability time for the i-th job on j-th machine.

V_{ij} = machine availability time for the i-th job on j-th machine.

3.4 Problem Formulation:

We have represented a job by an index i and a machine by an index j. Since an operation is specified entirely by the specification of the job and the machine involved, such an operation will be represented by a pair of indices (ij).

The indexing of jobs is arbitrary; it does not necessarily correspond to the sequence in which the jobs are performed on each machine or to the order in which the machines process each job. Since we may process the jobs on any machine in some sequence other than the natural job index, we shall designate the job as

$$i_1,\ i_2,\ \ldots,\ i_k,\ \ldots,\ i_N,$$

where N is the total number of jobs. For example, a particular job i_k indicates the job index which is in the sequence-position k.

The indexing of machines is in natural numbers. Put another way, the machines are numbered 1,2,..., M; and the operation of job i are correspondingly numbered i1, i2, ..., iM. Each job can be treated as if it had exactly m operations, for in cases where fewer operations exist, the corresponding processing times can be taken to be zero. If the j-th operation of any job precedes its k-th operation, then the machine required by the j-th operation has a lower number than the machine required by the k-th operation.

Because of exorbitant computational requirements for the flow shop scheduling problem, we have assumed that the order of jobs on all the machine is the same. Though this assumption may exclude optimal schedules but the saving in computational time will be enormous.

The job and machine availability times can be expressed mathematically by the following relationships:

$$u_{ij} = \text{Max}\,[u_{i,j-1}, v_{i,j-1}] + t_{i,j-1}\,; \qquad (1)$$

$$i = 1,2,\ldots, N$$

$$j = 2,3,\ldots, M$$

$$u_{i1} = 0; \qquad (2)$$

$$i = 1,2,\ldots, N$$

$$v_{i_k j} = \text{Max}\,[u_{i_{k-1} j}, v_{i_{k-1} j}] + t_{i_{k-1} j} \; ; \qquad (3)$$

$$k = 2, 3, \ldots, N$$

$$j = 1, 2, \ldots, M$$

$$v_{i_1 j} = 0 \; ; \qquad (4)$$

$$j = 1, 2, \ldots, M$$

Relations (1) and (2) imply that the job is available for operation only when the directly preceding operation on that job (technological constraint) has been completed. Relations (3) and (4) imply that the machine is available for the operation only when the directly-preceding operation on that machine (determined by the sequence of jobs on machines) has been completed.

Total time for job i on all the machines can be represented as,

$$TJOB_i = \sum_{j=1}^{M} t_{ij} \; ; \qquad (5)$$

$$i = 1, 2, \ldots, N \, .$$

Once u_{ij} and v_{ij} for all the operations for the generated sequence have been computed the schedule-time can be computed from the following relations:

$$T(S) = \max\,[u_{i_N M}, v_{i_N M}] + t_{i_N M} \qquad (6)$$

where,

T(S) is the schedule-time for sequence S.

The objective of our problem is to minimize T(S).

The determination of an optimum T(S) is a difficult work because of the high combinatorial nature of the flow-shop problem. Considerable amount of savings in computational effort can result from using heuristic approaches. However, a heuristic approach does not guarantee optimality. The effectiveness of a heuristic approach can be gauged by solving the problems for which optimal solutions are already available in the literature. In this thesis, two heuristic approaches have been developed and tested for the flow shop problems.

In the solution procedures of the proposed heuristic approaches, the above stated problem has been represented as connected linear graphs. We represent the network of operations by an oriented graph G = (X,Z), where X is the set of nodes, which are made up of ij $(i = 1,2,\ldots, N \; ; \; j = 1,2,\ldots, M)$ corresponding to the given operation with a starting node O and a terminal node T. Each node has a given processing time t_{ij}. Both the starting and terminal nodes have zero time contents. Z is the set of directed arcs. A directed arc (a,b) corresponds to the directly order relation $a \gg b$.

3.5 Proposed Heuristic Algorithms:

It is known that the processing times for each job on each machine has definite influence on the make-span, but the degree of contribution of various machines depends on their position in the machine ordering. This hint has been used to generate sequences.

In the present work two heuristic approaches have been developed. The first approach generates feasible sequences, which in number are equal to the total number of jobs in the given problem. All the generated sequences are different at least in sequence-position one. In other words, for job i (i = 1,2,..., N) to be in sequence-position one, one sequence is generated. The generated sequence, giving minimum schedule time is the suggested sequence for the problem by this approach. Six different strategies have been tried to generate sequences. The second approach generates only one schedule. For the generation of the schedule, three different strategies have been tried.

3.5.1 Heuristic Algorithm 1:

In essence, the algorithm consists of four general phases. The first phase involves the calculation of job and machine availability times for each node of the graph. In the second phase, the value of function $Z_s(i)$ which is used for sorting purposes is calculated. The subscript 's' stands for the

strategy adopted. The various strategies are discussed later. The third phase involves the generation of a feasible solution. It is done by sorting. At any iteration k, job k will be first in the sequence to be generated i.e. $i_1 = k$. The fourth phase is an evaluative phase. For all the generated sequences, schedule times are compared. The sequence giving minimum schedule time is selected.

Sortingis done according to non decreasing value of thefunctions. A state of conflict exists if more than one jobs have the same function value. The resolution ofsuch conflicts is accomplished by giving priority to the longest job among the jobs in conflict. If still conflict is there, it is resolved by giving priority to the job having least processing time for the first operation of that job.

Six different strategies have been attempted. All these strategies are based upon job and machine availability times of nodes, and can be represented by the following expressions.

$$Z_1(i) = \sum_{j=1}^{M} u_{ij} - TJOB_i \; ; \qquad (7)$$

$$i = 1,2,\ldots, N .$$

$$Z_2(i) = \sum_{j=1}^{M} \text{Max}\,(u_{ij}, v_{ij}) - TJOB_i \; ; \qquad (8)$$

$$i = 1,2,\ldots, N.$$

$$Z_3(i) = \sum_{j=1}^{M} |u_{ij} - v_{ij}| - t_{iII} ;$$

$$i = 1,2,\ldots, N \qquad (9)$$

$$Z_4(i) = \sum_{j=1}^{M} |u_{ij} - v_{ij}| - TJOB_i ;$$

$$i = 1,2,\ldots, N \qquad (10)$$

$$Z_5(i) = \sum_{j=1}^{M} |u_{ij} - v_{ij}| ;$$

$$i = 1,2,\ldots, N. \qquad (11)$$

$$Z_6(i) = \sum_{j=1}^{M} u_{ij} ;$$

$$i = 1,2,\ldots, N. \qquad (12)$$

Expression (12) represents the total job availability time of job i. The relationship given by expression (7) is obtained by subtraction of sum of processing times of operations of job i, from expression (12). This is done keeping in mind that effect of processing times has already been taken into account while calculating u_{ij}. Expression (11) yields total idle and job delay times for job i. The relationship given by expression (9) is obtained by subtraction of $TJOB_i$ from expression (11). In this strategy we have used the logic that the processing time of last operation of the job is not used for calculating job and machine availability times of the job i.

Expression (8) is obtained by subtraction of $TJOB_i$ from the total of processing start time of each node of job i.

Solution Procedure:

The step-by-step computational algorithm is given below:

Step 1: Construct the initial graph.

1.1 Construct N linear graphs, from the machine orderings such that $(0) \ll (i1) \ll (i2) \ll \ldots \ll (iM) \ll$ ($i = 1,2,\ldots, N$.

1.2 Assign processing times to each node. Processing time for node $ij = t_{ij}$; $i = 1,2,\ldots, N$; $j = 1,2,\ldots, M$, $t_0 = t_T = 0$.

1.3 Set $k = 0$.

Step 2: Calculate Job and machine availability times:

2.1 Set $k = k+1$

2.2 Set $i_1 = k$. For each machine, consider the operation of k-th job which directly precedes the operations of all other jobs. Calculate the job and machine availability time for each node.

2.3 Calculate the value of function for jobs i ($i = 1,2,\ldots, N$; $i \neq k$) according to the strategy adopted.

Step 3: Sort the Obtained Functions:

3.1 Sort the functions for jobs $i(i = 1, 2, \ldots, N;$ $i \neq k)$ in a non-decreasing order and accordingly assign the values to $i_2, i_3, \ldots, i_N$.

3.2 While sorting if there is some conflict, give priority to the job for which TJOB is maximum among the conflict set (set of jobs which are in conflict).

3.3 If still there are conflicts, give priority to the job i for which t_{i1} is minimum among the conflict set.

3.4 Order the jobs according to the sorted order of functions to obtain the sequence. Mathematically,

$$i_1 \ll i_2 \ll i_3 \ll \ldots \ll i_N.$$

3.5 Calculate schedule time for the sequence generated.

$$\text{Schedule time} = \max\,[u_{i_N M}, v_{i_N M}] + t_{i_N M}.$$

Step 4: Check if all necessary schedules have been generated.

If k = N, go to Step 5, otherwise, return to Step 2.

Step 5: Evaluate the best schedule.

Among the generated schedules the schedule which gives the minimum schedule time.

Illustrative Problem:

For the sake of illustration, a sample problem of 3 jobs and 4 machines has been selected from literature and solved using heuristic algorithm one employing strategy one. The processing time matrix for the problem which is taken from Brooks[12] is given below:

Processing Times

		Machines 1	2	3	4
Jobs	1	6	8	9	4
	2	1	3	9	6
	3	4	5	3	6

Step 1: The initial graph is constructed such that each linear graph is connected to initial and terminal nodes, as illustrated in Figure 1. Processing times are indicated on the nodes. The stateof iteration index (k) is set at zero.

Step 2: The iteration indexis increased to 1. For each machine the operation of k-th job which directly precedes the operation of all other jobs is identified (Fig. 2). Job and machine availability times are calculated for each node. The values of function $Z_1(i)$ for job i ($i = 1,2,3;\ i \neq k$) are:

$$Z_1(2) = 0 + 7 + 17 + 32 - 19 = 37$$

$$Z_1(3) = 0 + 10 + 19 + 26 - 18 = 37$$

(u_{ij}, v_{ij})

where u_{ij} is job availability time and v_{ij} is machine availability time.

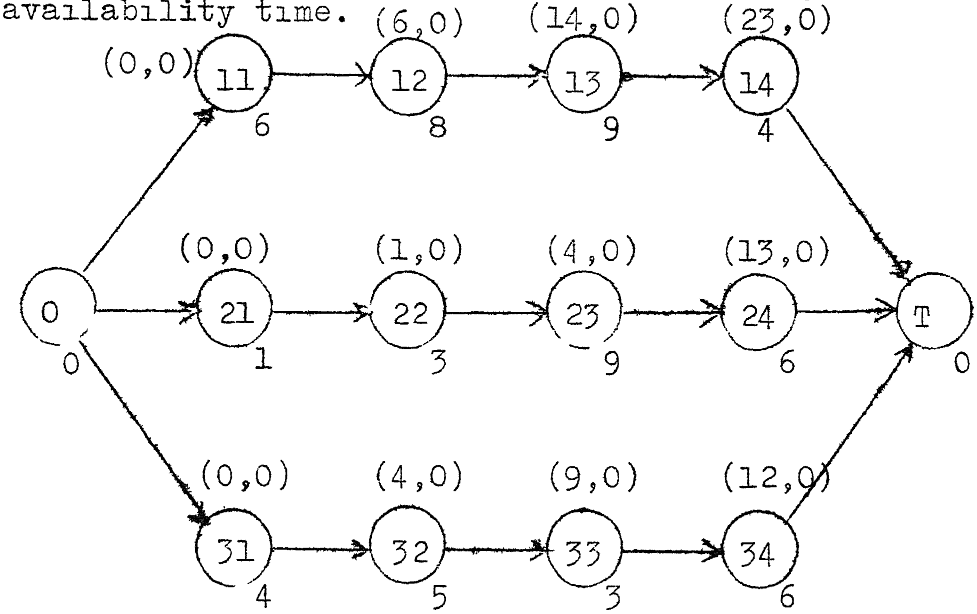

Figure 1

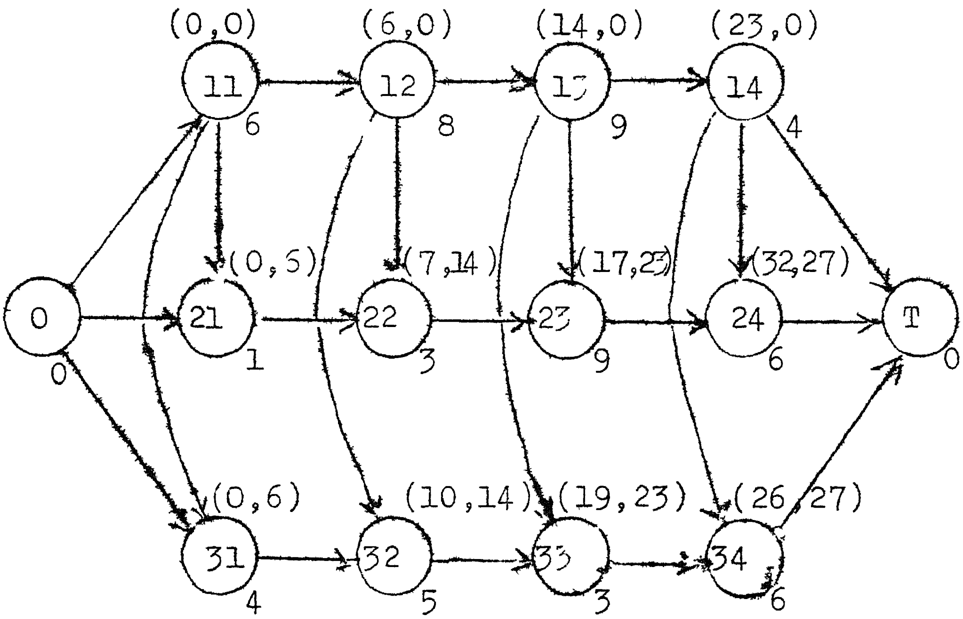

Figure 2

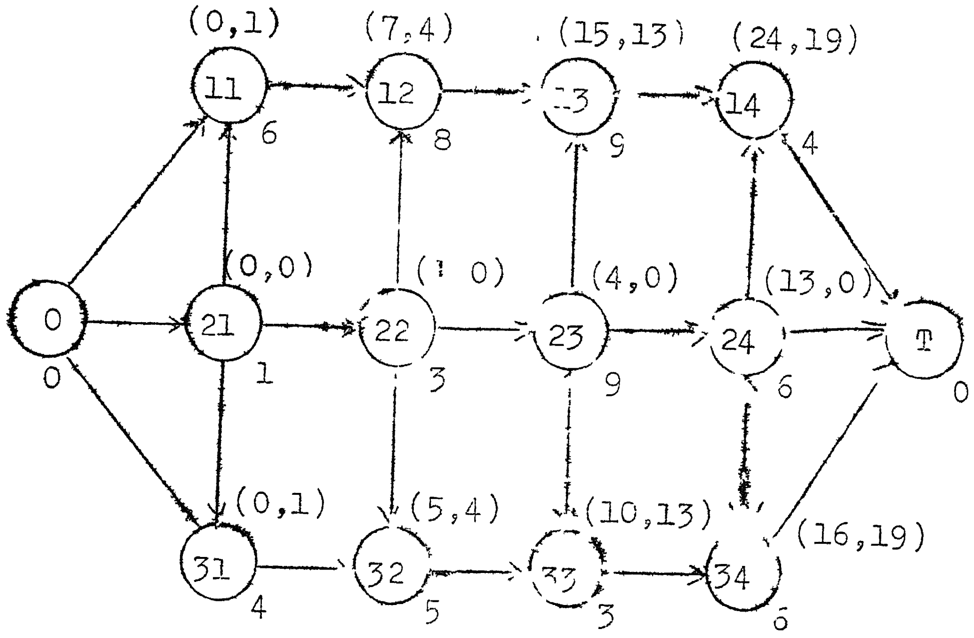

Figure 4

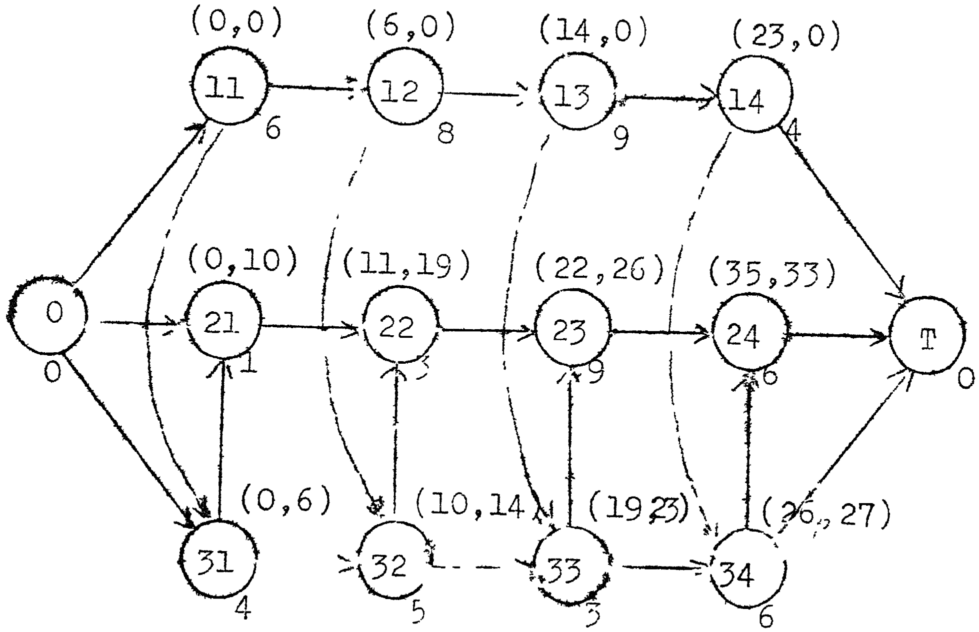

Figure 3

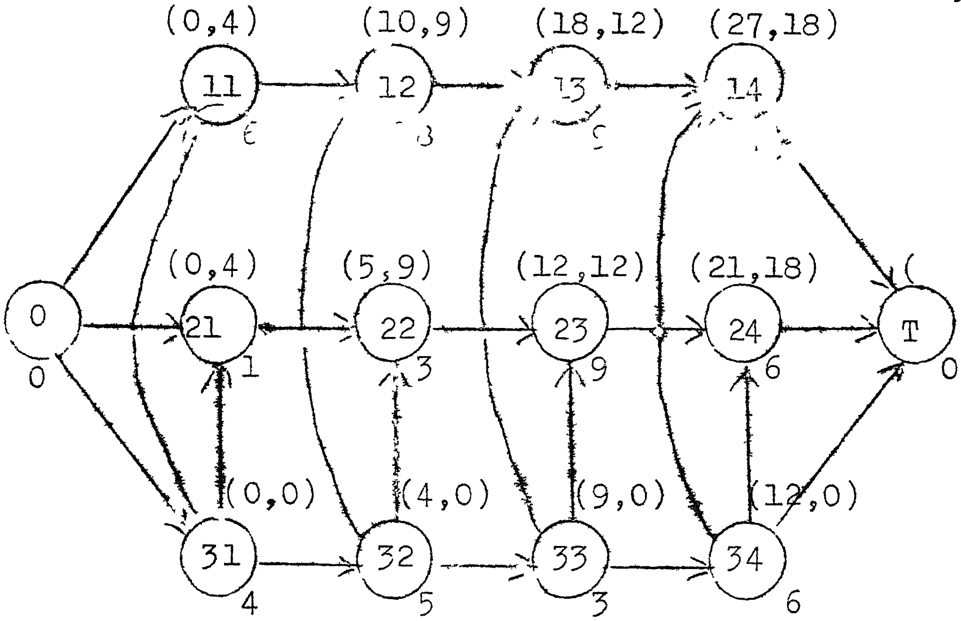

Figure 6

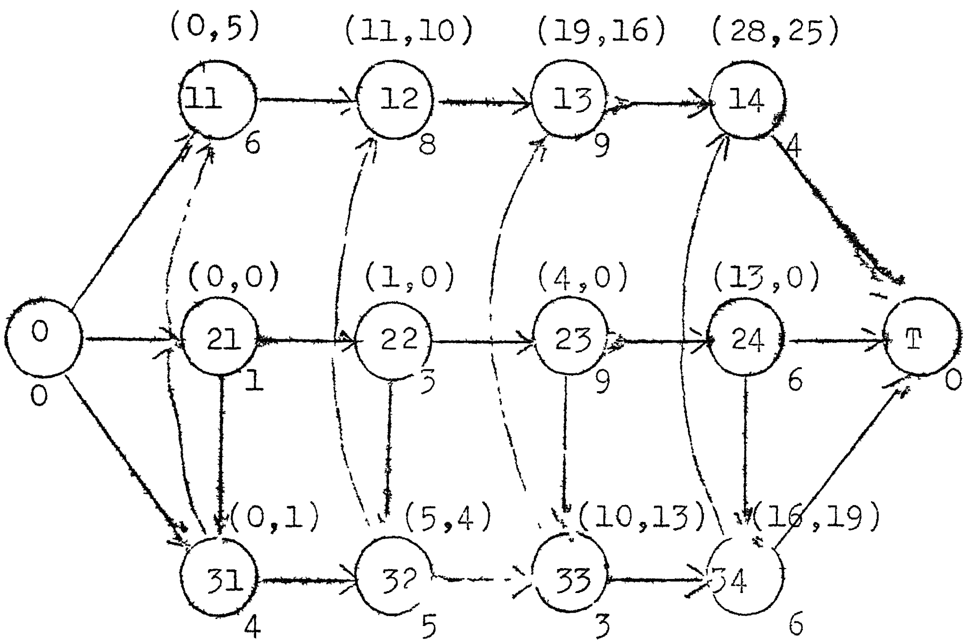

Figure 5

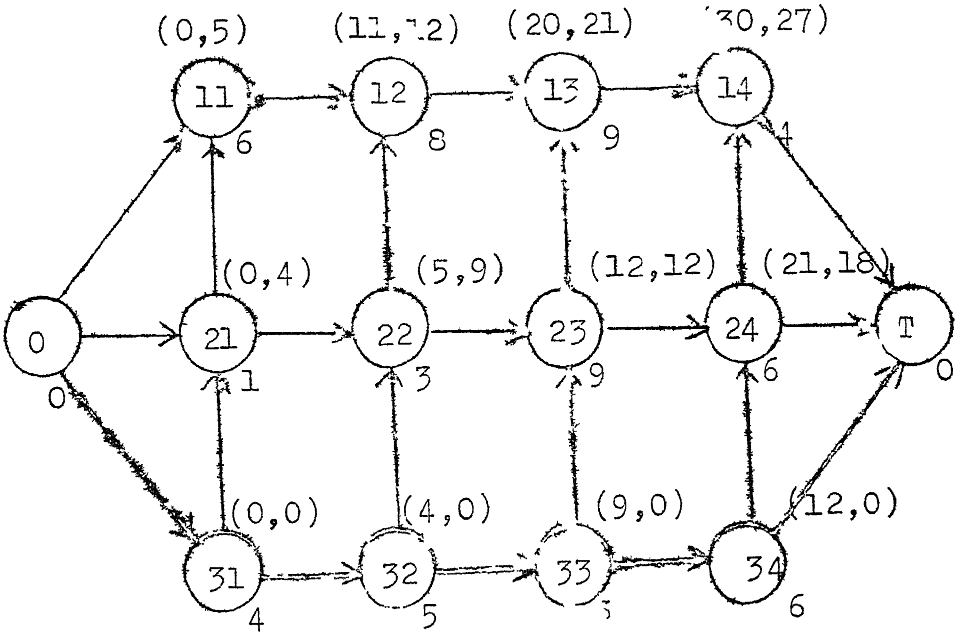
(0,5)
11
6
(11,12)
12
8
(20,21)
13
9
(30,27)
14
4
0
0
(0,4)
21
1
(5,9)
22
3
(12,12)
23
9
(21,18)
24
6
T
0
(0,0)
31
4
(4,0)
32
5
(9,0)
33
(12,0)
34
6

Figure 7

Step 3: The functions obtained in Step 2 are sorted out. Since the values of $Z_1(2)$ and $Z_1(3)$ are equal, there is a conflict. The conflict is resolved in favour of job 3 because $TJOB_3 > TJOB_2$. Thus, the obtained sequence is 1,3,2. For this sequence, job and machine availability times are calculated as shown in Fig.3. For the obtained sequence the schedule time is 41.

Step 4: A check is made to see if all the necessary schedules have been generated. The procedure continues through two more iterations such that the number of iterations become equal to the number of jobs. In the second iteration, sequence 2,3,1 is generated. This is shown in Fig.4. The schedule time for the sequence is calculated from Fig. 5, its value being 32.

Step 5: Among the three generated sequences, sequence 2,3,1 turns out to be best sequence giving a schedule time of 32. It needs to be pointed out that the best sequence obtained is also optimum solution to the problem.

3.5.2 Heuristic Algorithm II:

This algorithm conprises of three general phases. The first phase involves the calculation of job and machine availability times for each node of the graph. For a N-job problem, N graphs are generated. Each graph corresponds to a different starting job which precedes all other jobs. In the second phase the value of function (depending upon the adopted strategy) corresponding to all the N graphs is calculated. The third phase

involves the generation of sequence. It is done by sorting of functions. For the generated sequence schedule time is calculated.

Sorting is done according to non decreasing value of functions and their corresponding graph numbers are ordered. The final order of these graph numbers gives the desired sequence. A state of conflict exists if more than one graphs have their function values equal. The resolution of such conflicts is accomplished by giving priority to the longest job among the jobs in conflict. If still conflict exists, it is resolved by giving priority to the job having least processing time for the first operation of that job.

Three different strategies have been attempted. These strategies are strategy 7, strategy 8 and strategy 9 respectively. These strategies can be represented by the following relationships,

$$Z_7\ (k) = \sum_{i=1}^{N} \sum_{j=1}^{M} |u_{ij} - v_{ij}| \; ; \qquad k = 1,2,\ldots, N \tag{13}$$

$$Z_8\ (k) = \sum_{i=1}^{N} \sum_{j=1}^{M} \text{Max}\ (u_{ij}, v_{ij}); \qquad k = 1,2,\ldots, N \tag{14}$$

$$Z_9\ (k) = \sum_{i=1}^{N} \text{Max}\ (u_{iM}, v_{iM}) \; ; \qquad k = 1,2,\ldots, N \tag{15}$$

Expression (13) represents the total machine idle and job delay times created when job k precedes all other jobs. Expression (14) yields total processing start time. Expression (15) yields total of processing start time of last operation of each job.

Solution Procedure:

Following are the steps for solving the problem using the heuristic algorithm 2.

Step 1: Construct the initial graph. Construction of initial graph is same as in heuristic algorithm 1, however it has been repeated here.

1.1 Construct N linear graphs, from the machine orderings such that

$$(0) \ll (i1) \ll (i2) \ll \ldots \ll (iM) \ll (T) \quad ; \; i = 1,2,\ldots,N$$

1.2 Assign processing times to each node,

Processing time for node $ij = t_{ij}$; $i = 1,2,\ldots,N$; $j = 1,2,\ldots, M$. $t_0 = t_T = 0$.

1.3 Set $k = 0$.

Step 2: Calculate job and machine availability times.

2.1 Set $k = k + 1$

2.2 For each machine, consider the operation of k-th job which directly precedes the operations of all

other jobs. Calculate the job and machine availability time for each node.

2.3 Calculate the value of the function for graph k, according to the strategy adopted.

Step 3: Check of all the necessary graphs have been generated. If $k = N$, go to step 4, otherwise, return to step 2.

Step 4: Sort the obtained functions.

4.1 Sort the functions for graphs k ($k = 1,2,\ldots,N$) in a non decreasing order and accordingly assign the values to $i_1, i_2, \ldots, i_N$.

4.2 While sorting if there is some conflict, give priority to the job for which TJOB is maximum among the conflict set.

4.3 If still there are conflicts, give priority to the job i for which among the conflict set, for which t_{i1} is minimum.

4.4 Order the jobs according to the sorted order of functions to obtain the sequence. Mathematically,

$$i_1 \ll i_2 \ll i_3 \ll \cdots \ll i_N .$$

4.5 Calculate schedule time for the sequence generated. Schedule time = max $[u_{i_N M}, v_{i_N M}] + t_{i_N M}$.

Illustrative Problem:

The various steps of heuristic algorithm two are explained for the 3 job and 4 machine problem which was used for illustration of heuristic algorithm one. The values of the functions used for sorting are calculated using strategy 8. The steps are given below:

Step 1: Same as in the illustrative problem solved by heuristic one.

Step 2: The iteration index is increased to 1. For each machine, the operation of k-th job which directly precedes the operation of all other jobs is identified (Fig.2). Job and machine availability times are calculated for each node. The value of function $Z_8(k)$ is 82.

Step 3: A check is made to see if all the necessary networks have been generated. The procedure continues throughs two more iterations such that number of iterations becomes equal to the number of jobs. In the second iteration, network shown in Fig.4 is generated. The value of function $Z_8(2)$ turns out to be 37. The network for the third iteration is given in Fig. 6; the value of $Z_8(3)$ being 56.

Step 4: The functions obtained in Step 3 are sorted out. It yields the sequence 2,3,1. The job and machine availability times for this sequence are shown in Fig. 5. The schedule time for the sequence is 32.

3.6 Experimental Investigations:

3.6.1 Locally Optimal Schedules for Pseudorandom Problems:

Experiments have been conducted to select the best strategies for each of the two proposed heuristic algorithms. There are six strategies for heuristic algorithm one and three strategies for heuristic algorithm two. About 240 problems of various sizes have been generated and solved by both the heuristic algorithms. The various strategies for both the algorithms have been evaluated for all the problems. The problem size is varied from 5 jobs to 40 jobs and 3 machines to 20 machines. For each problem size, a set of 10 problems has been generated. The processing times of the jobs at various machines are obtained by generating integer pseudo-random numbers which come from a rectangular distribution between 0 and 99. The choice of the uniform distribution is in keeping with established practice in literature. A computer package' has been developed in FORTRAN IV for the IBM 7044/1401 computer system for testing the proposed algorithms.

All the strategies of heuristic algorithm one generate sequences equal to the number of jobs. Sequence giving minimum schedule time by each strategy of the heuristic algorithms is selected. Thus, for each problem we shall get nine sequences; one for each strategy of the heuristic algorithms. The sequence

which gives the minimum makespan value is termed as the best sequence. The performance of heuristics (and each strategy of heuristics) is evaluated based on the average relative error and number of times best sequence is obtained. The details of the two measures of evaluation are given in the following paragraphs.

Average Relative Error:

Let T_B and T_{S_i} be the makespan values of the best sequence and the sequence obtained by strategy 'i' of the proposed heuristic algorithms. Then the relative error, ER, is defined as:

$$ER_i = \frac{T_{S_i} - T_B}{T_B} \times 100 \qquad (16)$$

where, i = index of the strategy (i = 1,2,..., 9), and

$$T_B = \min_i \{ T_{S_i} \}$$

Then, the average error, AER, is computed for a set of N_1 problems of the same size.

$$AER = \frac{\sum_{j=1}^{N_1} ER_i}{N_1} \text{ ; for the same set of problems.} \qquad (17)$$

In this study, $N_1 = 10$.

The overall average, OAER, which is measure of overall effectiveness of the heuristic algorithm for all the problems considered is given by,

$$OAER = \frac{\sum_{k=1}^{N_2} AER}{N_2} \qquad (18)$$

where N_2 = number of problem sets.

In this study, $N_2 = 24$.

Number of Best Sequences Obtained:

For all the generated problems, which are 240 in number, the total number of best sequence obtained for each strategy of the heuristic algorithms are determined and compared.

The detailed results are presented in the next chapter. Based on the results obtained, it is found that heuristic one employing strategy one turns out to be best. However, the performance of strategy two for heuristic one is quite comparable with the best strategy. Therefore, the performances of strategies one and two of heuristic one have been tested on some of the problems available in the literature.

3.7.2 Performance on Previously Explored Problems:

The performances of heuristic one employing strategies one and two have been tested on 42 problems with known optimal

solution. The complete details of these problems are available in the literature. The performance is evaluated on the basis of relative error, which can be expressed as:

$$ER = \frac{T_{st} - T_0}{T_0} \times 100 \qquad (19)$$

where,

ER = Relative Error.

T_{st} = Makespan value obtained by heuristic one employing strategy t, t = 1 for strategy 1, t = 2 for strategy 2.

T_0 = Optimal makespan value.

The results are presented in the next chapter.

CHAPTER IV

INFERENCES AND SCOPE FOR FUTURE WORK

In this chapter the results of experimental investigations carried out to evaluate the performance of the proposed heuristic algorithms are presented. At the end of the chapter, a few possible avenues for further work are suggested.

4.1 Performance on Pseudorandom Problems:

Schedule times obtained using each strategy of both the heuristic algorithms for all the 240 problems were computed. The average relative error obtained using each strategy on 24 problem sets for heuristic algorithms one and two are presented in Tables 4.1 and 4.2 respectively. The average relative error for each problem size is calculated based on 10 problems for the problem size under consideration.

The results given in Tables 4.1 and 4.2 indicate that strategy one of heuristic one yields least overall relative error (1.578 percent) and generates maximum number of best sequences (87). However, the performance of strategy two of heuristic one is quite comparable with the performance of strategy one of the same heuristic. For strategy two, the overall relative error is 1.652 percent and the number of best sequences generated are 83.

4.2 Performance on Previously Explored Problems:

Fourty two problems with known optimal solutions are solved by the proposed heuristic algorithm one employing strategies one and two. The obtained schedule times and relative errors are presented in Table 4.3. The results of Table 4.3 indicate that heuristic one using strategy one generated optimal sequences for 22 problems and resulted in an average relative error of 2.467. The average relative error and the number of best sequences obtained employing strategy two are 2.983 percent and 14 respectively. Therefore, it is concluded that the performance of strategy one (heuristic one) is the best of all the strategies considered for the two heuristic algorithms.

Schedule times obtained by heuristic one using strategy one for four problems available in literature with unknown optimal solutions are presented in Table 4.4. In the following paragraphs results obtained by the proposed heuristic algorithm one employing strategy one are compared with the existing algorithms in the literature.

A 10-job, 3-machine problem, which according to Ignall and Schrage[29] is the hardest sequencing problem was solved by the proposed heuristic algorithm one. Ignall and Schrage report that the optimal sequence is 123...8910 with a makespan value 66 for this problem. The best sequence generated by the

proposed heuristic one results with a makespan value of 68. However, Ignall and Schrage's branch and bound method required the evaluation of 2570 nodes for this problem. By generating only ten out of |10 possible sequences, the proposed heuristic one with strategy one produces sequence which is very close to the optimal sequence.

Campbell, Dudek and Smith[14] have solved a 8-job, 7-machine problem by their heuristic algorithm and generated six sequences. The sixth sequence found had the least schedule time (595). They have also mentioned that optimal makespan for this problem is 584. Hence. their algorithm resulted in a relative error of 1.9 percent. The proposed heuristic one with strategy one generated optimal sequence giving total makespan 584.

Baker[6] has solved a 5-Job, 3-Machine problem by the algorithms of Palmer[40], J.N.D. Gupta[22] and Campbell, Dudek and Smith[14] and reported the obtained schedule time as 37, 36 and 35 by the respective algorithms. The proposed heuristic algorithm one resulted in a makespan of 34 which is less than the other three algorithms mentioned above.

A 8-Job, 5-Machine problem from Subrahmanyam, J.V.[45] on solving by proposed heuristic algorithm one resulted into a makespan of 604. Subrahmanyam has reported that the algorithm proposed by Palmer[40], J.N.D. Gupta[22], and Campbell, Dudek and Smith[14] develop sequences giving makespan 615, 661 and 607 respectively.

Campbell, Dudek and Smith[14] have mentioned that with their algorithm, computation times were insensitive to the number of machines. On IBM 7044 they took 1.806 minute to solve 60-job problem. The proposed algorithm took about 1.90 minutes to solve 100-job, 10-machine problem. For most of the problems the proposed algorithm would take computational time comparable to Campbell, Dudek and Smith algorithm.

For a 6-Job, 3-Machine problem from Gupta[24], the proposed heuristic algorithm one results into makespan of 53. Gupta[24] has reported that, his job pairing algorithm[24] and min-idle rule[23] result in makespan of 54 and 53 units respectively. However, optimal solution to this problem results in the makespan of 49 units.

Six problems used by Palmer (given in reference 40, Table 1) for testing his heuristic-algorithm were solved by the proposed algorithm. Palmer has mentioned that his slop index method gives total of makespan (for all the six problems), 19 above the total of the optima. However, proposed heuristic one generates schedule, giving total makespan to be 13, above the total of the optima for the six problems. This again proves the superiority of the proposed algorithm over palmer's slop index algorithm.

Giglio and Wagner[20] have solved six problems (given in reference 20, Table II) by a heuristic algorithm based on

Johnson's method. They found that by their method, for the six problems, the average ratio (optimal value/Johnson solution value) was 0.95. These problems were solved by the proposed algorithm and resulted in the average ratio (optimal value/proposed solution value) of 0.968. It shows that for these six problems, the proposed heuristic algorithm one worked efficiently as compared to the heuristic algorithm (extension of Johnson's algorithm) proposed by Giglio and Wagner.

The performance of the proposed heuristic one has been further tested on two Sample problems of size 100-job, 10-machines, and 20-jobs, 10 machines, used by Heller[28] in his random-sampling study. In the experiment on 100-job, 10-machine problem, in 3000 trials, Heller obtained sample schedule time and sample variance as 656.81 and 20.80 respectively. He found some schedule times at integer values from the schedule time of 606 to the schedule time of 707. Near the tail of the distribution he failed to find some schedule times. By the proposed algorithm one with strategy one, we are able to find a schedule time of 547, and only 100 sequences were generated. In the second experiment on 20-job, 10-machine problem, in 12,000 trials Heller obtained sample schedule time of 169.93 and sample variance of 1.24. For the same problem, Ashour[5] took 237.2 secs. on the UNIVAC 1108 computer for its solution and generated a schedule giving schedule time as 164. The

proposed heuristic algorithm one gives makespan of 158 and took only 3 secs. (Execution time) to get this solution on IBM 7044 computer.

4.3 Conclusions:

The important features of the present study can be summarized as follows:

i) The basic concept of the proposed algorithms embodies the sorting of functions which are based on job and machine availability times. The number of sequences generated by the first algorithm are equal to the number of jobs, where as second algorithm generates only one sequence.

ii) One of the primary advantage of the proposed heuristic algorithms is their ability to generate good quality solutions with relatively less computational effort and storage requirements.

iii) The heuristic algorithm one with strategy one is found to be superior over all other proposed strategies of heuristic algorithm one and two. However, among the strategies of heuristic two strategy eight results in comparatively better results. It is worth mentioning here that heuristic two comparatively takes less computational time as compared to heuristic one.

iv) For most of the problems attempted in thesis, heuristic algorithm one employing strategy one turns out to be superior to the heuristics cited in the literature.

4.4 Scope for Further Work:

The area of scheduling is extremely rich and large number of real life industrial problems still remain unsolved. A few possible avenues for further research based on the work presented in this thesis are given below:

i) More powerful strategies for both the proposed algorithms could be developed which would enrich the solution quality capability of the algorithms, yet not alter the computational characteristics drastically.

ii) Heuristic algorithm two offers scope for extension to handle job shop scheduling problem.

iii) The proposed heuristic algorithms are based on the criterion of minimization of makespan time. Another important criterion for scheduling pertains to the minimization of average flow time. Using the basic philosophy of the proposed heuristic algorithms, strategies can be developed for the criterion of minimization of average flow time.

Table 4.1: Performance Comparison of Strategies of Heuristic one for Pseudorandom Problems.

No.	Problem size	Average Relative Error for Strategies					
		1	2	3	4	5	6
1	10 x 8	3.300	0.685	1.430	1.900	2.470	4.820
2	15 x 15	1.705	1.958	2.000	2.180	7.680	1.250
3	20 x 15	0.964	1.010	3.770	1.010	5.61	4.62
4	20 x 5	1.023	0.694	2.050	0.958	4.32	4.08
5	20 x 12	2.16	0.628	1.88	2.07	1.895	2.78
6	30 x 10	1.008	1.592	1.883	1.285	1.495	2.165
7	40 x 10	0.862	0.757	1.665	1.537	2.030	2.818
8	10 x 3	0.034	1.560	2.125	2.865	4.36	3.378
9	15 x 5	1.71	1.543	2.83	1.668	3.54	4.88
10	20 x 10	1.054	1.723	2.16	1.38	2.722	2.725
11	15 x 10	0.882	1.261	2.74	1.832	3.54	2.52
12	20 x 8	1.676	1.808	3.21	1.082	3.422	6.340
13	15 x 8	0.726	1.682	2.081	1.642	2.078	2.54
14	10 x 5	3.435	2.665	3.60	3.805	6.28	5.67
15	20 x 20	1.712	1.462	1.22	1.203	1.695	0.973
16	8 x 3	0.82	2.368	2.65	3.055	5.28	5.64
17	10 x 9	1.635	1.328	3.17	3.56	4.26	4.4
18	10 x 6	2.035	1.442	2.495	1.647	3.78	4.30
19	10 x 7	2.79	3.065	3.46	2.935	4.28	4.92
20	10 x 4	0.159	1.796	3.84	3.01	8.55	10.74
21	5 x 8	3.14	1.855	1.83	2.295	3.72	4.29
22	10 x 15	1.715	1.580	2.466	2.110	2.320	3.650
23	5 x 10	2.10	3.10	3.22	3.060	3.20	2.48
24	10 x 20	1.975	2.055	2.445	1.850	2.325	3.001
Overall Average Relative error		1.578	1.652	2.51	2.08	3.782	3.955
No. of best sequences obtained		87	83	54	57	38	28

Table 4.2 Performance Comparison of Strategies of Heuristic Two for Pseudo-Random Problems.

No.	Problem Size	Average Relative Error for Strategies		
		7	8	9
1	10 x 8	10.740	7.010	11.710
2	15 x 15	10.370	5.600	6.180
3	20 x 5	12.450	6.880	7.020
4	25 x 5	8.570	4.83	9.13
5	20 x 12	10.5	3.35	8.72
6	30 x 10	10.689	3.48	4.82
7	40 x 10	11.78	3.175	4.65
8	10 x 3	7.24	5.999	9.45
9	15 x 5	10.58	7.140	10.89
10	20 x 10	7.7	2.87	6.85
11	15 x 10	10.3	4.42	6.13
12	20 x 8	10.490	5.78	10.02
13	15 x 8	11.76	4.61	5.76
14	10 x 5	6.05	6.72	9.72
15	20 x 20	8.82	2.3	4.92
16	8 x 3	5.17	8.37	10.3
17	10 x 9	10.5	6.96	11.35
18	10 x 6	12.55	5.68	7.32
19	10 x 7	8.58	5.28	9.56
20	10 x 4	11.45	13.48	17.82
21	5 x 8	5.73	7.35	7.42
22	10 x 15	8.41	5.28	7.67
23	5 x 10	5.37	5.26	6.98
24	10 x 20	10.59	3.92	5.83
Overall Average relative error		9.43	5.65	8.34
No. of best sequences obtained		13	30	8

Table 4.3: Performance Comparison of Strategies one and two of Heuristic one for Published Sample Problems with Known Optimal Solutions.

No.	Problem size	Ref. No.	Optimum Schedule time	Strategy One		Strategy Two	
				Obtained schedule time	Relative Error	Obtained schedule time	Relative Error
1	10 x 3	29	66	68	3.0	67	1.5
2	5 x 3	43	1078	1086	00.8	1093	1.5
3	4 x 3	37	56	58	3.6	58	3.6
4	4 x 3	6	39	45	15.4	39	0.0
5	4 x 3	37	28	28	0.0	31	10.7
6	3 x 4	12	32	32	0.0	34	6.24
7	4 x 5	22	33	38	15.15	38	15.15
8	8 x 7	14	584	584	0.0	584	0.0
9	6 x 3	18	460	480	4.35	460	0.0
10	6 x 3	20	68	68	0.0	70	2.95
11	6 x 3	20	76	81	6.58	79	3.95
12	4 x 3	47	22	23	4.55	23	4.55
13	6 x 3	24	49	53	8.17	55	12.25
14	6 x 5	13	97	97	0.0	99	2.06
15	7 x 3	40	69	72	4.35	72	4.35
16	8 x 3	40	81	83	2.47	83	2.47
17	6 x 3	40	57	59	3.51	57	0.0
18	9 x 3	40	83	83	0.0	84	1.21
19	4 x 3	40	54	54	0.0	54	0.0
20	5 x 3	40	74	80	8.12	78	5.41
21	6 x 3	38	46	48	4.35	46	0.0
22	5 x 4	42	135	135	0.0	138	2.22
23	4 x 4	44	155	155	0.0	155	0.0
24	4 x 3	37	56	56	0.0	56	0.0
25	5 x 3	37	63	64	1.59	64	1.59
26	6 x 3	37	64	68	6.25	68	6.25
27	4 x 3	37	34	34	0.0	34	0.0
28	5 x 3	37	42	42	0.0	42	0.0
29	4 x 3	37	44	44	0.0	46	4.56
30	5 x 3	37	59	59	0.0	61	3.39
31	6 x 3	37	63	63	0.0	65	3.17
32	7 x 3	37	74	74	0.0	76	2.7
33	8 x 3	37	77	77	0.0	78	1.3
34	4 x 3	37	54	56	3.71	56	3.71
35	5 x 3	37	59	59	0.0	59	0.0
36	6 x 3	37	69	71	2.9	71	2.9
37	7 x 3	37	76	78	2.63	78	2.63
38	3 x 3	37	27	27	0.0	30	11.1
39	5 x 2	37	47	48	2.13	48	2.13
40	3 x 2	37	19	19	0.0	19	0.0
41	3 x 2	10	31	31	0.0	31	0.0
42	2 x 2	21	10	10	0.0	10	0.0

Table 4.4: Performance Evaluation of Heuristic One Employing Strategy One for Published Sample Problems with Unknown Optimal Solutions.

No.	Problem size	Reference No.	Obtained Schedule time
1	5 x 3	6	34
2	20 x 10	28	158
3	100 x 10	28	547
4	8 x 5	45	604

REFERENCES

1. Ashour, S., 'An Experimental Investigation and Comparative Evaluation of Flow-Shop Scheduling Techniques', Operations Research, Vol. 18, No. 3, 1970, pp. 541-549.

2. Ashour, S., and A.R. Char, 'Computational Experience on Zero-One Programming Approach to Various Combinatorial Problems, 'Journal of Operations Research Society of Japan, Vol. 13, No. 2, 1970, pp. 78-108.

3. Ashour, S., 'Sequencing Theory', Lecture Notes in Economics and Mathematical Systems, Springer-Verlag, New York, 1972.

4. Ashour, S., 'A Statistical Analysis of Production Scheduling Systems', Journal of Operations Research Society of Japan, Vol. 12, No. 2, 1970, pp. 65-68.

5. Ashour, S. and R.G. Parker, 'An Out-of-Kilter Approach for Machine Sequence Problems', Symposium on the Theory of Scheduling and its Applications, Edited by S.E. Elmaghraby, Springer-Verlag, New York, 1973.

6. Baker, K.R., 'Introduction to Sequencing and Scheduling', John Wiley and Sons, Inc., New York, 1974.

7. Baker, K.R., 'A Comparative Study of Flow-Shop Algorithms', Operations Research, Vol. 23, No. 1, 1975, pp. 62-73.

8. Baker, K.R., 'An Elimination Method for the Flow-Sho Problem', Operations Research, Vol. 23, No. 1, 1975, pp. 159-162.

9. Bakshi, M.S. and S.R. Arora, 'The Machine Sequencing Problem', Management Science, Vol. 16, No. 4, 1969, pp. B 247-263.

10. Balas, E., 'Machine Sequencing via Disjunctive Graphs: An Implicit Enumeration Algorithm', Operations Research, Vol. 17, 1969, pp. 941-957.

11. Bowman, E.H., 'The Scheduling-Sequencing Problem', Operations Research, Vol. 7, No. 5, 1959, pp. 621-624.

12. Brooks, H. and Charles R. White,' An Algorithm for Finding Optimal or Near-Optimal Solution to the Production Scheduling Problem', Journal of Industrial Engineering, Vol. 16, No. 1, 1965, pp. 34-40.

13. Brown, A.P.G., and Z.A. Lomnicki, 'Some Applications of the Branch and Bound Algorithm to the Machine Scheduling Problem', Operational Research Quarterly, Vol. 17, No. 2, 1966, pp. 173-186.

14. Campbell, H.G., Richard A. Dudek and Milton L. Smith, 'A Heuristic Algorithm for the n-Job, m-Machine Sequencing Problem', Management Science, Vol. 16, No. 2, 1970, pp. B630-637.

15. Charlton, John M., 'A Method of Solution for General Machine Scheduling Problem', Operations Research, Vol.18, No. 4, 1970, pp. 689-707.

16. Conway, R.W., W.L. Maxwell and L.W. Miller,'Theory of Scheduling', Addison-Wesley Publishing Company, Reading, Massachusetts, 1967.

17. Day, J.E., and Michael, P. Hottenstein,' Review of Sequencing Research', Naval Research Logistic Quarterly, Vol. 17, No. 1, 1970, pp. 11-39.

18. Dudek, R.A. and O.F. Teuton, Jr., 'Development of M-Stage Decision Rule for Scheduling n-Jobs Through M-Machines', Operations Research, Vol. 12, No. 3, 1964, pp. 471-447.

19. Giffler, B. and G.L. Thompson', Algorithm for Solving Production Scheduling Problems', Operations Research, Vol. 8, No. 4, 1960, pp. 487-503.

20. Giglio, R.J., and H.M. Wagner', Approximations to the Three-Machine Scheduling Problems', Operations Research, Vol. 12, No. 3, 1964, pp. 305-324.

21. Greenberg, H., 'Solution of the General Scheduling Problem', Operations Research, Vol. 16, 1968, pp. 353.

22. Gupta, J.N.D., 'A Functional Heuristic Algorithm for the Flow-Shop Scheduling Problems', Operational Research Quarterly, Vol. 22, No. 1, 1971, pp. 39-48.

23. Gupta, J.N.D., 'Heuristic Rules for n x M Flow-Shop Scheduling Problem', Opsearch, Vol. 5, No. 3, 1968, pp. 165-170.

24. Gupta, J.N.D., and Albert R. May Kut,' Heuristic Algorithms for Scheduling n Jobs in a Flow-Shop', Journal of Operations Research Society of Japan, Vol. 16, No. 3, 1973, pp. 131-150.

25. Gupta, J.N.D., 'M-Stage Flow-Shop Scheduling by Branch and Bound', Opsearch, Vol. 7, No. 1, 1970, pp. 37-43.

26. Gupta, J.N.D., 'Analysis of a Combinatorial Approach to Flow Shop Scheduling Problems', Operational Research Quarterly, Vol. 22, No. 2(ii), 1975, pp. 431-440.

27. Heller, J. and G. Logemann', An Algorithm for the Construction and Evaluation of Feasible Schedules', Management Science, Vol. 8, No. 2, 1962, pp. 168-183.

28. Heller, J., 'Some Numerical Experiments for M x J Flow-Shop and its Decision-Theoretical Aspects', Operations Research, Vol. 8, No. 2, 1960, pp. 178-184.

29. Ignall, E. and L. Schrage, 'Application of the Branch and Bound Technique to some Flow-Shop Scheduling Problems', Operations Research, Vol. 13, No. 3, 1965, pp. 400-412.

30. Johnson, S.M., 'Optimal Two and Three Stage Production Schedules with Set-up Times Included', Naval Research Logistic Quarterly, Vol. 1, No. 1, 1954, pp.61-68.

31. Karush, W., 'A Counter Example to a Proposed Algorithm for Optimal Sequencing of Jobs', Operations Research, Vol. 13, No. 2, 1965, pp. 323-325.

32. Krone, M.J. and Kenneth Steiglitz,' Heuristic Programming Solution of a Flow-Shop Scheduling Problem', Operations Research, Vol. 22, No. 3, 1974, pp. 629-638.

33. Little, J.D.C., K.G. Murthy, D.W. Swenny and C. Karel 'An Algorithm for the Travelling Salesman Problem', Operations Research, Vol. 11, No. 6, 1963, pp. 972-989.

34. Lomnicki, Z.A., 'A Branch and Bound Algorithm for the Exact Solution of Three Machine Scheduling Problem', Operational Research Quarterly, Vol. 16, No. 1, 1965, pp. 89-100.

35. Manne, A.S., 'On the Job Shop Scheduling Problem', Operations Research, Vol. 8, No. 2, 1960, pp. 219-223.

36. McMohan, C.B. and P.G. Burton, 'Flow-Shop Scheduling with the Branch and Bound Method', Operations Research, Vol.15, No. 3, 1967, pp. 473-481.

37. Muth, J.F., and G.L. Thompson, Eds., Industrial Scheduling, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1963.

38. Nabeshima, I., 'On the Bound of Makespans and its Application in M Machine Scheduling Problem', Journal of Operations Research Society of Japan, Vol. 9, Nos. 3 and 4, 1967, pp. 98-136.

39. Nabeshima, I., 'The Order of n items Processed on m Machines', Journal of Operations Research Society of Japan, Vol. 16, No. 3, 1973, pp. 163-185.

40. Palmer, D.S., 'Sequencing Job Through a Multi-Stage Process in the Minimum Total Time - A Quick Method of obtaining a Near Optimum', Operational Research Quarterly, Vol. 5, No. 2, 1973, pp. 142-151.

41. Reddy, S.S. and C.V. Ramamoorthy, 'On the Flow-Shop Sequencing Problem with No Wait in Process', Operational Research Quarterly, Vol. 23, No. 3, 1972, pp. 323-331.

42. Shore, B., Operations Management, McGraw-Hill Book Co. New York.

43. Smith, R.D. and R.A. Dudek, 'A General Algorithm for Solution of the n-Job, M-machine Sequencing Problem of the Flow-Shop', Operations Research, Vol. 15, No. 1, 1967, pp. 71-82.

44. Smith, M.L., S.S. Panwalkar and R.A. Dudek, ' Flowshop Sequencing Problem with Ordered Processing time Matrices', Management Science, Vol. 21, No. 5, 1975, pp. 544-549.

45. Subrahmanyam, J.V., 'Heuristics In Scheduling', Unpublished M.Tech. Thesis, IIT Kanpur, 1973.

46. Subrahmanyam, V., Heuristic Programming Approaches to Static Flowshop Sequencing Problem', Unpublished M.Tech. Thesis, IIT-Kanpur, 1975.

47. Szwarc Wlodzimerz, 'Optimal Elimination Methods in the m x n Flow-shop Scheduling Problem', Operations Research, Vol. 21, No. 6, 1973, pp. 1250-1259.

48. Wagner, H.M., 'An Integer Programming Model for Machine Scheduling', Naval Research Logistic Quarterly, Vol. 6, No. 2, 1959, pp.131-140.

49. Wismer, D.A., 'Solution of the Flow-Shop Scheduling Problem with No Intermediate Queues', Operations Research, Vol. 20, No. 3, 1972, pp. 689-697.

APPENDIX

Definitions of Basic Terms:

Job:

A job is a unit of a product or a batch of identical units, that must be processed on certain machines. Alternative names are task, commodity.

Machine:

A machine is a single device capable of performing a certain process. An alternative name is a facility, processor, or work center.

Operation:

An operation is an elemental task to be performed on a job by a particular machine. An operation is specified entirely by the job and the machine involved. An alternative name is activity.

Make-Span Time:

The make-span time (total elapsed time; schedule time or maximum flow time) is defined as the total elapsed time between the start of the first job at the first machine and the total time of the last job in the sequence at the last machine.

Processing Time:

A processing time is the length of time which is required for the completion of an operation on a particular machine. An alternative name is an operation time, or running time. The processing times for all operations may be combined in a matrix referred to as a processing time matrix.

Machine Ordering:

A machine ordering is an arrangement of a set of machines through which a particular job is to be performed, depending primarily on the technological requirements. An alternative name is a routing, ordering, or technological ordering.

Job Sequencing:

A job sequencing is an arrangement of a set of jobs to be processed on a single machine.

Sequence:

A sequence is a set of job sequencings, each of which is assigned to each of the machines. The sequence specifies the arrangement of the operations comprising all jobs on all machines. A sequence does not provide the time at which the operations are performed nor the existence of idle times between various operations.